

[Get started](#)[Log in](#)

Douglas Matoso

Using Git (and GitHub) on Windows

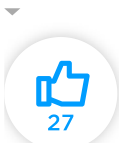
Douglas Matoso

Apr 20, 2017 • 6 Min read • 28,281 Views

Apr 20, 2017 • 6 Min read • 28,281 Views

Software Engineeri...

Summary



- [Summary](#)
- [Installing Git](#)
- [Creating a local repository](#)
- [Sharing your code on GitHub](#)
- [Creating your first remote repository](#)
- [Conclusion](#)
- [Top ^](#)

Summary

This guide will show how to install Git on Windows, create a local repository and then pushing it to GitHub.

Installing Git

Download the installer for Windows from the [Git official site](#).

Execute the downloaded file. In the page **Select Components** you can leave the options at their defaults.

The **Windows Explorer integration > Context menu entries** option allows opening the Git command

We use cookies to make interactions with our websites and services easy and meaningful. For more information about the cookies we use or to find out how you can disable cookies, [click here](#).

[Disable cookies](#)[Accept cookies and close this message](#)

Bash Here. The last option is also interesting, that it installs a better font for all console windows.

Note: Git for Windows comes with it's own command prompt (Git Bash), that besides git commands it has some useful Unix commands (and it looks better than Windows default prompt).

On the next screen (**Adjusting your path environment**), I choose the most conservative option: **Use Git Bash only**. This will make git commands available only in Git Bash, and will not alter your PATH variable. Click **Next**.

Another important setting: line endings. As you may know, Windows and Unix systems (Linux, Mac) have different formats of line breaks on text files. if you write a file with Windows line breaks, another person may have problems opening that file in Linux, and vice-versa. This settings allows you to normalize this.

I prefer choosing the second option (**Checkout as-is, commit Unix-style line endings**), that won't change the line breaks when the file arrives, but will convert them to Unix-style when you commit. That way you don't risk commit Windows-style line breaks and everything is kept on Unix-style. (Don't worry, even though you are in Windows, most of text editors can read Unix line breaks just fine)

After that one more **Next, Finish** and Git is installed! \o/

Let's test it? Create a folder, right click it and choose **Git Bash Here**.

Before anything else, let's inform Git who you are, so your commits can be identified. Enter the commands, replace the quoted data with your real name and e-mail: (press Enter after each one)

```
1  git config --global user.name "Firstname  
2  git config --global user.email "your_email"
```

Now let's initialize a Git repository on this folder:

```
1  git init
```

See that (master) on the command line? It tells you the current branch you are in a Git repository. The **master** branch is the main branch on every Git repo.

Now let's add a new file and commit it. Look at the command sequence (press Enter after each one):

```
1  touch test.txt  
2  git add .  
3  git commit -m "First commit"
```

First we create an empty text file (you can create the file any way you like, not necessarily with the `touch` command). Then we `add` all new and modified files to the Git index (we tell Git which files we want to commit on the next commit). And

Sharing your code on GitHub

Cool! You have a Git repo in your machine, but how about sharing your code on GitHub?

Initial setup

If you don't have a GitHub account yet, go to <http://github.com> and create one. It's free.

After you signup and login, let's add a SSH key so GitHub can link your account with this computer. That way it won't have to ask for your password on every commit.

On Git Bash enter the command:

```
1 ssh-keygen -t rsa -C "your_email@example."
```

Use the same email you registered at GitHub.

On the next question press Enter to choose the default value.

Now it will ask for a password. Enter a password (this is NOT your GitHub password). When it asks for a confirmation, enter the password again. Now enter the command:

```
1 notepad ~/.ssh/id_rsa.pub
```

To open on Notepad the file that was created.

On GitHub go to **Settings** and then **SSH and GPG**

this computer and in the field **Key** paste all the contents of the file `id_rsa.pub`.

Be careful to copy and paste all the contents of the file, beginning at "`ssh-ras ...`" up to your email (including it). Click **Add SSH key**.

Let's check if everything is ok. On Git Bash enter:

```
1  ssh -T git@github.com
```

It will ask if you want to connect to a remote machine. Type `yes` and press Enter. Next it will ask for a password. Enter the password you used on the `ssh-keygen` command.

If you see a message like:

```
1  Hi user! You've successfully authenticate
```



Then everything is correct!

Creating your first remote repository

On GitHub let's create a new repository (button **New repository** on your dashboard). Enter a name. It should not have spaces or special characters, as it will be part of the URL of your new repo. You can leave the rest of the options at their defaults.

You will be taken to the main page of your repository, that doesn't have any files yet

On Git Bash (on the folder of your local repository) enter:

```
1 git remote add origin git@github.com:user
```

Note that `user/repo_name` must be entered the same way they appear in your repository URL, like:

https://github.com/user/repo_name

Now, to send your files to GitHub, enter:

```
1 git push origin master
```

Inform the password of the SSH key if it asks.

Reload the page of your repo on GitHub and you should see your committed files.

Conclusion

Even though Git originated on Linux (did you know that it was created by Linus Torvalds, the same guy who created Linux?), developers on all system can benefit from it. Git is an excellent SCM (source code management) system, widely adopted, and the open-source community on GitHub is vibrant! You can find code for pretty much anything you want, contribute with other developers and share your own solutions.

See you next!

We use cookies to make interactions with our websites and services easy and meaningful. For more information about the cookies we use or to find out how you can disable cookies, [click here](#).

[Disable cookies](#)

[Accept cookies and close this message](#)



Test your skills. Learn something new. Get help. Repeat. [Start a FREE 10-day trial](#)

ABOUT

SOLUTIONS

PLATFORM

SUPPORT

COMPANY

CONTACT

EVENTS

- | | | | |
|--------------------------|-----------------------|--------------|-------------------------|
| Business | Browse library | Help center | Partners |
| Personal | Role IQ | Integrations | PluralsightOne. |
| Small business | Skill IQ | IP whitelist | Customer stories |
| Academic | Iris | | Careers |
| Federal government | Paths | | Teach |
| State & local government | Projects | | Blog |
| | Interactive Courses | | Newsroom |
| | Guides | | Pluralsight engineering |
| | Authors | | Affiliate program |
| | Mobile apps | | |
| | Professional Services | | Subscribe |